BAB 2

LANDASAN TEORI

2.1 Teori-teori Dasar/ Umum

2.1.1 Teknologi

Teknologi berasal dari literatur yunani, yaitu *technologia* yang diperoleh dari asal kata *techne* yang bermakna wacana seni. Pada abad ke-17, maknanya adalah pembahasan sistematis atas 'seni terapan'. Kemudian, pada abad ke-20, maknanya diperluas menjadi metode dan teknik non-material. Menurut Capra, sebagian besar definisi teknologi sekarang lebih menekankan hubungannya dengan sains. Capra juga mendefinisikan teknologi sebagai kumpulan alat, aturan, dan prosedur yang merupakan penerapan pengetahuan ilmiah terhadap suatu pekerjaan tertentu.

Menurut Djoyohadikusumo, teknologi berkaitan erat dengan sains dan perekayasaan. Dengan kata lain, teknologi mengandung dua dimensi yaitu *science* dan *engineering* yang saling berkaitan satu sama lain.

2.1.2 Informasi

Informasi berasal dari bahasa Perancis kuno, yaitu *informacion* yang diambil dari bahasa Latin, *informationem,* yang berarti "garis besar, konsep, ide" . Informasi juga dapat diartikan sebagai data yang telah diolah menjadi bentuk yang lebih berguna dan berarti bagi yang menerimanya.

Pengertian informasi menurut Williams & Sawyer (2011, p25) adalah data yang telah dirangkum atau dimanipulasi dalam bentuk lain untuk tujuan pengambilan keputusan.

2.1.3 Teknologi Informasi

Teknologi informasi atau sering disebut IT memiliki banyak sekali pengertian, berikut ini adalah pengertian teknologi informasi menurut para pakar yang dapat dijadikan acuan :

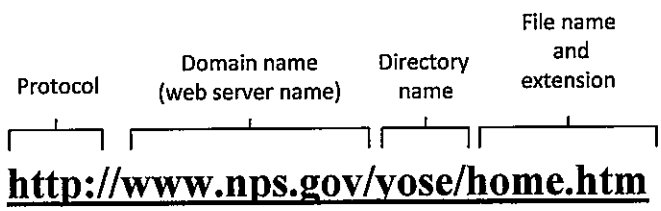
1. Menurut Williams & Sawyer (2011, p4), teknologi informasi adalah teknologi yang membantu manusia dalam membuat, mengubah, menyimpan, mengomunikasikan dan atau menyebarkan informasi. Teknologi informasi menggabungkan komputasi dengan link komunikasi yang berkecepatan tinggi untuk menyampaikan data, suara dan video.
2. Menurut Turban, Rainer dan Potter (2005, p5), teknologi informasi secara umum adalah sekumpulan dari sistem komputer yang digunakan perusahaan. Hampir semua perusahaan, swasta *(private)* dan terbuka *(public),* dalam kebanyakan industry, menggunakan teknologi informasi untuk mendukung operasinya. Alasan dari luasnya pemakaian TI ini adalah TI telah menjadi fasilitator utama dari aktivitas bisnis di dunia saat ini.

2.1.4 Internet

Internet (Williams & Sawyer, p18) merupakan jaringan komputer di seluruh dunia yang menghubungkan ratusan bahkan ribuan jaringan yang lebih kecil, misalnya jaringan pendidikan, komersial, ilmiah, nirlaba sebaik jaringan individu. Dengan adanya internet, jutaan orang di seluruh dunia dapat berbagi berbagai jenis informasi dan layanan.

Selain itu, Williams dan Sawyer juga menguraikan beberapa istilah yang berkaitan dengan internet, antara lain :

1. URL (*Uniform Resource Locator*) adalah string karakter yang menunjuk ke sebuah bagian informasi tertentu di mana saja pada web. URL terdiri dari empat bagian, yaitu web protocol, nama domain atau web server, direktori pada server dan nama file tanpa direktori.



Gambar 2.1 URL

2. *World Wide Web* adalah sistem interkoneksi server internet yang mendukung dokumen berformat multimedia, seperti teks suara, foto dan video. Web memungkinkan pengguna komputer untuk mengakses informasi antar sistem di seluruh dunia menggunakan URL.

3. HTML *(Hypertext Markup Language)* adalah sekumpulan perintah khusus ("tag" atau "markup") yang dipakai untuk menentukan struktur, bentuk, dan link pada dokumen ke dokumen multimedia lain di web. HTML merupakan dasar pembuatan halaman web.

2.1.5 Delapan Aturan Emas dalam desain *interface* (*Eight Golden Rules)*

Dalam mendesain *user-interface* yang baik, dibutuhkan delapan aturan yang disebut dengan "*Eight Golden Rules" (*Shneiderman, 2005, p74), antara lain :

1. Berusaha untuk konsisten

   Konsistensi dilakukan pada urutan tindakan, prompt, menu, layar bantuan, warna yang konsisten, tata letak, kapitalisasi, dan font.

2. Menyediakan *usability* universal

   Menyediakan fitur bagi para pemula, seperti penjelasan dan ada pula fitur bagi para ahli untuk meningkatkan kecepatan interaksi, seperti *shortcut.* Aturan ini dapat digunakan untuk memperkaya desain *interface* dan meningkatkan kualitas sistem.

3. Memberikan umpan balik yang informatif

   Untuk setiap aksi pengguna, harus ada umpan balik sistem. Untuk tindakan yang sering dilakukan dan tidak terlalu penting, dapat diberikan umpan balik yang sederhana. Tetapi ketika tindakan merupakan hal yang penting, maka umpan balik sebaiknya lebih

substansial. Misalnya muncul suatu suara ketika salah menekan tombol pada waktu input data atau muncul pesan kesalahannya.

4. Merancang dialog yang memberikan penutupan (keadaan akhir)

   Urutan tindakan sebaiknya diorganisir dalam suatu kelompok dengan bagian awal, tengah, dan akhir. Umpan balik yang informatif akan memberikan indikasi bahwa cara yang dilakukan sudah benar dan dapat mempersiapkan kelompok tindakan berikutnya.

5. Memberikan pencegahan kesalahan dan penanganan kesalahan yang sederhana

   Sedapat mungkin sistem dirancang sehingga pengguna tidak dapat melakukan *error*. Jika kesalahan terjadi, sistem dapat mendeteksi kesalahan dengan cepat dan memberikan mekanisme yang sedehana dan mudah dipahami untuk penanganan kesalahan.

6. Memungkinkan pembalikan aksi yang mudah

   Hal ini dapat mengurangi kekuatiran pengguna karena pengguna mengetahui kesalahan yang dilakukan dapat dibatalkan sehingga pengguna tidak takut untuk mengekplorasi pilihan-pilihan lain.

7. Mendukung pusat kendali internal

   Pengguna ingin menjadi pengontrol sistem dan sistem akan merespon tindakan yang dilakukan pengguna. Sebaiknya sistem dirancang sedemikan rupa sehingga pengguna menjadi inisiator daripada responden.

8. Mengurangi beban ingatan jangka pendek

   Keterbatasan ingatan manusia membutuhkan tampilan yang sederhana atau banyak tampilan halaman yang sebaiknya disatukan, serta diberikan cukup waktu pelatihan untuk kode, mnemonic, dan urutan tindakan.

2.1.6 Basis Data

Menurut Mannino (2001, p7), basis data adalah bahasa untuk mendefinisikan entitas, hubungan, batasan integritas dan hak otorisasi. Sedangkan menurut Connolly dan Begg (2005, p15), basis data merupakan suatu kumpulan data yang terbagi atas data yang berhubungan secara logis dan deskripsi yang dirancang untuk memenuhi kebutuhan informasi suatu organisasi. Connolly dan Begg mengungkapkan bahwa basis data adalah tempat penyimpanan tunggal yang besar bagi data yang dapat digunakan secara bersamaan oleh banyak departemen dan pengguna. Basis data tidak hanya menyipan data operasional organisasi tetapi juga menyimpan deskripsi data yang disebut dengan meta data. Basis data menggambarkan entitas, atribut dan hubungan logis antara entitas. Dan dengan kata lain, basis data menyimpan data yang terkait secara logis.

2.1.7 DBMS

DBMS adalah sekumpulan perangkat lunak yang mendukung penciptaan, penggunaan dan pemeliharaan basis data. Definisi ini

dikemukakan oleh Mannino (2001, p7). Sedangkan menurut Connolly dan Begg (2005, p16), DBMS (*Database Management System*) adalah sebuah sistem perangkat lunak yang memungkinkan pengguna untuk menetapkan, membuat, memelihara dan mengontrol akses ke basis data. DBMS menyediakan beberapa fasilitas sebagai berikut :

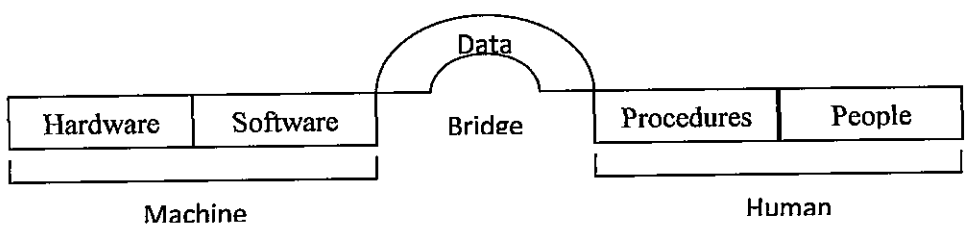
1. *Data Definition Language (DDL)* adalah suatu bahasa yang mengizinkan atau memungkinkan pengguna untuk mendeskripsikan dan memberi nama pada entitas, atribut dan relasi yang dibutuhkan untuk sebuah aplikasi termasuk batasan-batasan keamanan dan integritasnya. DDL digunakan untuk menetapkan suatu skema atau memodifikasi data yang sudah ada. DDL tidak bisa digunakan untuk memanipulasi data.
2. *Data Manipulation Language (DML)* adalah suatu bahasa yang menyediakan operasi dasar manipulasi data pada data yang terdapat di dalam basis data. DML memungkinkan pengguna untuk *insert, update, delete,* dan mendapatkan kembali data dari basis data. Bagian dari DML yang melibatkan pengambilan data disebut dengan *query language*. *Query language* dapat didefinisikan sebagai bahasa tingkat tinggi yang digunakan untuk memenuhi berbagai permintaan untuk pengambilan data dalam basis data. Ada dua tipe DML :
    a. DML prosedural : bahasa yang memungkinkan pengguna untuk memberi instruksi kepada sistem mengenai data yang dibutuhkan dan cara pemanggilannya.

b. DML non-prosedural : bahasa yang memungkinkan pengguna untuk menentukan data yang dibutuhkan dengan menyebutkan spesifikasinya tanpa menspesifikasikan bagaimana cara mendapatkannya.

3. Menyediakan akses kontrol ke basis data, seperti :
   a. Sistem keamanan, yang mencegah pengguna tidak sah mengakses basis data.
   b. Sistem integritas, yang mempertahankan konsistensi data yang disimpan.
   c. Sistem kendali konkurensi, yang memungkinkan akses bersama pada basis data.
   d. Sistem kontrol pemulihan, yang mengembalikan basis data ke keadaan konsisten sebelumnya setelah kegagalan perangkat keras atau perangkat lunak.
   e. Catalog *user-accessible*, yang berisi deksripsi data dalam basis data.

2.1.8 Komponen DBMS

Menurut Connolly dan Begg (2005, p18-21), komponen basis data terdiri dari lima komponen, antara lain :



Gambar 2.2 Komponen DBMS

1. *Hardware*

   Hardware dapat berkisar dari komputer pribadi tunggal, *mainframe* tunggal, hingga sebuah jaringan komputer. *Hardware* tertentu tergantung pada kebutuhan organisasi dan DBMS yang dibutuhkan. Sebuah DBMS membutuhkan jumlah minimum memori utama dan ruang disk untuk bekerja, tetapi konfigurasi minimum tidak memberikan performa yang handal.

2. *Software*

   Komponen ini terdiri dari *software* DBMS itu sendiri dan program aplikasi bersama sistem operasi termasuk *software* jaringan di DBMS yang sedang digunakan di jaringan. Biasanya, program aplikasi ditulis dalam bahasa pemrograman, seperti C, C++, Java, Visual Basic, COBOL, Fortran, Ada, Pascal atau SQL.

3. Data

   Data merupakan komponen yang terpenting di dalam DBMS. Data berfungsi sebagai jembatan antara komponen mesin (*hardware* dan *software*) dan komponen manusia (prosedur dan manusia). Basis data berisi data operasional dan *meta data.*

4. Prosedur

   Prosedur mengacu pada petunjuk dan aturan yang mengatur desain dan penggunaan basis data. Penggunaan sistem dan petugas yang mengelola basis data memerlukan dokumentasi prosedur tentang bagaimana menggunakan dan menjalankan sistem. Instruksi tersebut seperti :

a. Bagaimana cara masuk ke dalam DBMS
b. Bagaimana menggunakan fasilitas DBMS atau program aplikasi
c. Bagaimana memulai dan menghentikan DBMS
d. Bagaimana menangani kesalahan *hardware* dan *software*
e. Bagaimana cara mengubah struktur tabel, mengatur ulang basis data, meningkatkan kinerja dan arsip data untuk penyimpanan sekunder

5. Manusia

Komponen manusia terdiri dari : Data Administrator, Basis data Administrator, pengembang aplikasi, basis data desainer, dan *end-users* (klien).

2.1.9 Keuntungan dan Kerugian DBMS

Menurut Connolly dan Begg (2005, p26-p30), DBMS memiliki keuntungan dan kerugian sebagai berikut :

2.1.9.1 Keuntungan DBMS

a. Kontrol terhadap pengulangan data (*data redundancy)*

Basis data berupaya untuk menghilangkan redundansi dengan mengintegrasikan file sehingga beberapa salinan dari data yang sama tidak disimpan. Namun, basis data tidak sepenuhnya menghilangkan redundansi, tetapi mengendalikan jumlah redundansi yang melekat dalam basis data.

b. Konsistensi data

Dengam menghilangkan atau mengendalikan redundansi, kita mengurangi resiko terjadinya ketidakkonsistenan yang terjadi. Jika *item* data disimpan hanya sekali dalam basis data, maka setiap *update* nilainya harus dilakukan hanya sekali dan nilai baru tersebut harus tersedia dengan segera untuk semua pengguna. Tetapi jika *item* data disimpan lebih dari sekali, sistem dapat memastikan bahwa semua salinan dari *item* tersebut tetap disimpan secara konsisten.

c. Banyak informasi yang didapat dari data yang sama

Dengan data operasional yang terintegrasi, hal ini memungkinkan bagi organisasi untuk mendapatkan informasi tambahan dari data yang sama.

d. Pembagian data (*sharing of data*)

Basis data termasuk bagian dari keseluruhan organisasi dan dapat dibagikan oleh semua pelanggan yang berotoritas. Dalam hal ini, banyak pengguna membagikan lebih banyak data.

e. Meningkatkan integritas data

Integritas basis data mengacu pada validitas dan konsistenci data yang disimpan. Integritas biasanya diekspresikan dalam istilah batasan, yang berupa aturan konsisten yang tidak boleh dilanggar oleh basis data. Integrasi

memungkinkan DBA untuk menjelaskan dan memungkinkan DBMS untuk membuat batasan integritas.

f. Meningkatkan keamanan data

Keamanan basis data yaitu melindungi basis data dari pengguna yang tak berotoritas. Hal ini dapat dilakukan dengan menggunakan sistem *username* dan *password* untuk mengidentifikasikan orang yang berotoritas pada basis data mungkin dibatasi oleh jenis operasi seperti pengambilan, *insert*, *update* dan *delete* data.

g. Menetapkan standarisasi

Integrasi memungkinkan DBA untuk mendefinisikan dan membuat standar yang diperlukan. Standar ini termasuk standar departemen, organisasi, nasional, atau internasional dalam hal format data, untuk memfasilitasi pertukaran data antara sistem, ketetapan penamaan, standar dokumentasi, prosedur *update*, dan aturan akses.

h. Skala ekonomi

Dengan menyatukan semua data operasional organisasi ke dalam satu basis data dan menciptakan sekelompok aplikasi yang bekerja pada satu sumber data yang dapat menghasilkan penghematan biaya. Dalam kasus ini, anggaran yang biasanya akan dialokasikan untuk masing-masing departemen untuk pengembangan dan pemeliharaan

sistem berbasis file yang dapat dikombinasikan dan mengakibatkan total biaya yang lebih rendah.

i. Menyeimbangkam konflik kebutuhan

Setiap pengguna memiliki kebutuhan yang mungkin bertentangan dengan kebutuhan pengguna lain. Sejak basis data dikendalikan oleh DBA, DBA dapat membuat keputusan yang berkaitan dengan penggunaan desain dan penggunaan operasional basis data yang menyediakan penggunaan terbaik dari sumber daya bagi keseluruhan organisasi.

j. Meningkatkan kemampuan akses dan respon pada data

Dengan pengintegrasian data yang melintasi batasan departemen dapat secara langsung diakses pada pengguna akhir. Hal ini dapat menyediakan sebuah sistem dengan lebih banyak fungsi seperti fungsi untuk menyediakan layanan lebih baik pada pengguna akhir atau klien organisasi. Banyak DBMS menyediakan *query language* yang memungkinkan pengguna untuk menanyakan pertanyaan *ad hoc* dan memperoleh informasi yang diperlukan dengan segera pada terminal mereka, tanpa memerlukan *programmer* menulis beberapa *software* untuk mengubah informasi ini dari basis data.

2.1.9.2 Kerugian DBMS

a. Kompleksitas

Penyediaan fungsi yang kita harapkan dari DBMS yang baik membuat DBMS menjadi sebuah *software* yang sangat kompleks. Perancangan dan pengembangan basis data, DA dan DBA, serta pengguna akhir harus memahami fungsi tersebut untuk mendapatkan banyak keuntungan dari DBMS ini.

b. Ukuran

Kompleksitas dan fungsi yang luas membuat DBMS menjadi sebuah *software* yang sangat besar, memerlukan banyak ruang ruang *hardisk* dan jumlah memori yang besar untuk berjalan secara efisien.

c. Biaya dari suatu DBMS

Biaya DBMS bervariasi, tergantung pada lingkungan dan fungsi yang disediakan. Terdapat juga biaya pemeliharaan tahunan yang juga dimasukkan dalam daftar harga DBMS.

d. Biaya perangkat keras tambahan

Kebutuhan tempat penyimpanan bagi DBMS dan basis data sangat memerlukan pembelian tempat penyimpanan tambahan. Selanjutnya, untuk mencapai performa yang diperlukan, mungkin diperlukan untuk membeli mesin yang lebih besar dan sebagainya. Hal ini tentu

memerlukan tambahan biaya yang tidak sedikit. Tergantung pada spesifikasi *hardware* yang diperlukan.

e. Biaya konversi

Dalam beberapa situasi, biaya DBMS dan *hardware* tambahan mungkin tidak signifikan apabila dibandingkan dengan biaya konversi aplikasi yang ada untuk dijalankan pada DBMS yang baru. Biaya ini juga termasuk biaya pelatihan para *staff* untuk menggunakan sistem baru, dan menjalankan sistem.

f. Performa

Sistem berbasis file ditulis untuk aplikasi yang spesifik, seperti *invoice*. Sebagai akibatnya, kinerja secara umum sangat baik. Namun, DBMS ditulis untuk melayani banyak aplikasi dan efeknya adalah bahwa beberapa aplikasi tidak dapat berjalan secepat yang dulu.

2.1.10 Fungsi DBMS

Menurut Connolly dan Begg (2005,p), fungsi DBMS adalah sebagai berikut :

1. Penyimpanan, pengambilan dan peng-*update*-an data
2. Katalog *user-accessible*
3. Mendukung transaksi
4. Layanan kendali konkurensi
5. Layanan perbaikan

6. Layanan autorisasi
7. Mendukung bagi komunikasi data
8. Layanan integritas
9. Layanan peningkatan keterbatasan data
10. Layanan utilitas

2.1.11 Bahasa Basis Data

Menurut Connolly dan Begg (2005, p39-40), sebuah data *sublanguage* terdiri dari dua bagian yaitu *Data Definition Language* (DDL) dan *Data Manipulation Language* (DML). DDL digunakan untuk menentukan skema basis data dan DML digunakan untuk membaca dan meng-*update* basis data. Keduanya disebut data *sublanguage* karena mereka tidak membangun semua kebutuhan pemrograman komputer seperti pernyataan kondisi dan iteratif yang digunakan oleh beberapa bahasa pemrograman tingkat tinggi lainnya.

a. Data Definition Language

Data Definition Language menurut Connolly dan Begg (2005,p40) adalah suatu bahasa yang memperoleh DBA atau pengguna untuk mendeskripsikan dan memberi nama entitas, atribut, dan relasi yang diperlukan untuk aplikasi. DDL berfungsi untuk mengubah suatu data menjadi data yang berguna bagi pengguna. Beberapa statement DDL menurut Connolly dan Begg (2002,p167):

1. *Create Table*

   Untuk membuat table dengan mendefinisikan tipe data untuk tiap kolom.

2. *Alter table*

   Untuk menambah atau membuang kolom dan konstrain.

3. *Drop Table*

   Untuk membuang dan menghapus tabel beserta semua data yang terkait didalamnya.

4. *Create Index*

   Untuk membuat index pada suatu tabel.

5. *Drop Index*

   Untuk membuang atau menghapus index yang sudah dibuat sebelumnya.

b. *Data Manipulation Language*

Menurut Connolly dan Begg (2005,p41), DML adalah suatu bahasa yang menyediakan sekumpulan operasi yang diinginkan untuk mendukung operasi manipulasi data utama pada data yang diperoleh dalam basis data. DML menyediakan operasi dasar manipulasi data pada data yang ada dalam basis data, yaitu:

- penyisipan data baru ke dalam basis data *(insertion)*
- mengubah atau memodifikasi data yang disimpan dalam basis data *(modify)*
- pemanggilan data yang ada dalam basis data *(retrieve)*

- menghapus data dari basis data (*delete)*

Menurut Connolly dan Begg, DML dibagi menjadi dua tipe yang berbeda, yaitu :

1. Prosedural DML

   Prosedural DML adalah suatu bahasa yang memungkinkan pengguna untuk memberitahu sistem mengenai data apa yang dibutuhkan dan bagaimana cara memanggilnya (*retrieve*).

2. Non Prosedural DML

   Non Prosedural DML adalah suatu bahasa yang memungkinkan pengguna untuk menentukan data apa yang dibutuhkan dengan menyebutkan spesifikasinya tanpa menspesifikasikan bagaimana cara mendapatkannya.

2.1.12 Struktur Data Relasional

Menurut Connolly dan Begg (2005, p72-p74), struktur data relasional terbagi menjadi beberapa bagian, yaitu :

1. Relasi : merupakan sebuah tabel dengan kolom dan baris. Digunakan untuk menyimpan informasi tentang objek yang digambarkan dalan basis data.
2. Atribut : merupakan nama kolom dari sebuah relasi. Atribut dapat ditampilkan dalam berbagai perintah dan dalam relasi yang sama sehingga menyampaikan arti yang sama.

3. *Domain* : merupakan sekelompok nilai yang diijinkan bagi satu atau lebih atribut. Setiap atribut dalam relasi didefinisikan pada sebuah *domain. Domain* dapat berbeda bagi setiap atribut, dua atau lebih atribut dapat didefinisikan pada *domain* yang sama. Konsep *domain* sangat penting karena memungkinkan pengguna menjelaskan arti dan sumber nilai yang ada pada atribut.
4. *Tuple* : merupakan baris dari sebuah relasi. *Tuple* dapat disebut *intention* jika struktur relasi, *domain* serta batasan-batasan lainnya pada nilai yang mungkin bersifat tetap, namun sebaliknya jika relasi berubah setiap waktu ini disebut *extension.*
5. *Degree* : merupakan jumlah atribut yang terdapat dalam relasi. Jika relasi mempunyai satu atribut akan mempunyai derajat satu yang disebut relasi *unary/* satu *tuple.*
6. *Cardinality* : merupakan jumlah *tuple* yang terdapat dalam relasi. Merupakan *property* dari *extension* relasi dan ditentukan dari *instance* tertentu.
7. Basis data relasional : merupakan kumpulan dari relasi yang ternormalisasi dengan nama relasi yang berbeda.

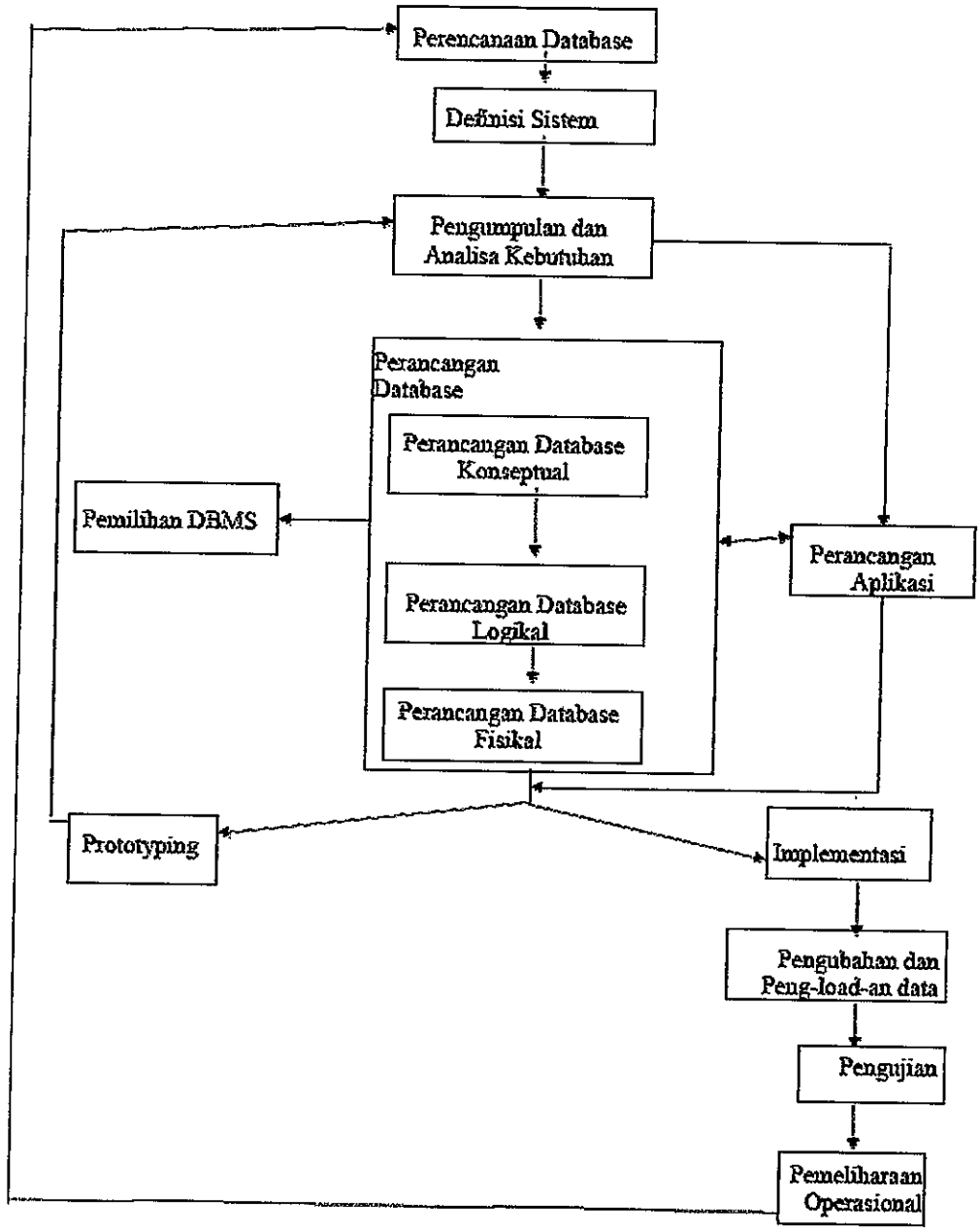
2.1.13 *Relational Keys*

Menurut Connolly dan Begg (2005, p78-79), *relational keys* dibagi menjadi beberapa jenis, yaitu :

1. *Superkey* : merupakan sebuah atribut atau sekelompok atribut yang mengidentifikasi secara unik *tuple* dalam relasi. *Superkey* yang mudah diidentifikasi adalah yang hanya berisi jumlah minimum atribut yang diperlukan.
2. *Candidate key* : merupakan *superkey* dalam relasi. *Candidate key* (K, bagi sebuah relasi (R) mempunyai dua sifat, yaitu :
    a. Keunikan : dalam setiap *tuple* dari R, nilai dari K secara unik mengidentifikasi *tuple* tersebut.
    b. *Irreducibility* : tidak ada subset yang sesuai dari K yang mempunyai keunikan sifat ketika sebuah *key* terdiri dari lebih dari satu atribut yang disebut sebagai *composite key.*
3. *Primary key* : merupakan *candidate key* yang terpilih untuk identifikasi *tuple* secara unik dalam satu relasi. Sementara *candidate key* yang tak terpilih sebagai *primary key* disebut *alternate key.*
4. *Foreign key* : merupakan sebuah atau sekelompok atribut dalam relasi yang dibandingkan dengan *candidate key* pada beberapa relasi.

2.1.14 Siklus Hidup Aplikasi Basis Data

Menurut Connolly dan Begg (2005, p273-p293), siklus hidup aplikasi basis data terdiri dari sepuluh tahapan, yaitu :



Gambar 2.3 Siklus Hidup Basis Data

2.1.15 UML (*Unified Modelling Language*)

Menurut Connolly & Begg (2005, p836) UML umumnya didefinisikan sebagai bahasa standar untuk menentukan, membangun, memvisualisasi, dan mendokumentasikan *artifact* dari sistem perangkat lunak. UML menyediakan bahasa umum untuk menggambarkan model perangkat lunak dan tidak menentukan metodologi tertentu, tetapi lebih fleksibel dan disesuaikan agar sesuai dengan setiap pendekatan serta dapat digunakan dalam hubungannya dengan berbagai siklus hidup perangkat lunak dan pengembangan proses.

Tujuan utama dalam desain UML adalah untuk:

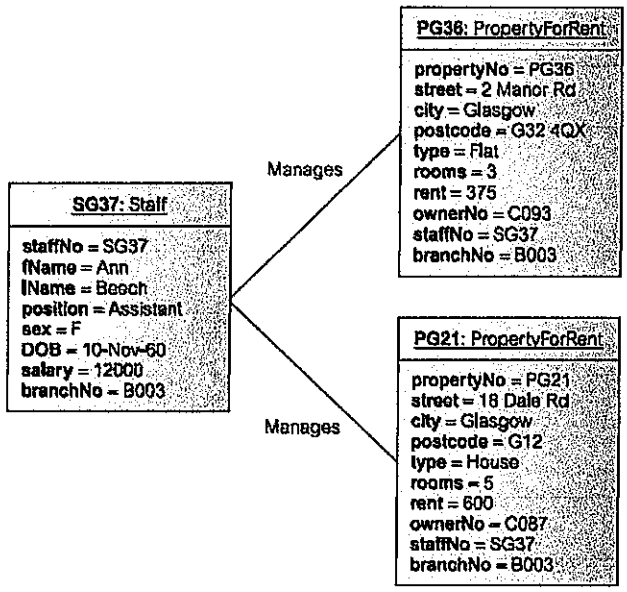
a. Menyediakan pengguna dengan *ready-to-use*, pemodelan bahasa visual yang ekspresif sehingga dapat mengembangkan dan menukarkan model yang bermakna.
b. Menyediakan mekanisme yang diperpanjang dan dispesialisasi untuk memperluas konsep-konsep inti. Untuk contoh, UML menyediakan stereotip yang memungkinkan elemen-elemen baru yang akan didefinisikan dengan memperluas dan menyempurnakan semantik elemen yang ada.
c. Memberikan dasar formal untuk memahami bahasa pemodelan.
d. Mendukung konsep pengembangan *high-level*, seperti kolaborasi, kerangka, pola, dan komponen.

UML mendefinisikan sejumlah diagram yang dapat dibagi menjadi dua kategori sebagai berikut:

1. Struktural diagram, menggambarkan hubungan yang statis antar komponen yang meliputi:
   a. *class diagrams*
   b. *object diagrams*
   c. *component diagrams*
   d. *deployment diagrams*
2. *Behavioral diagrams* (perilaku diagram), menggambarkan hubungan yang dinamis antar komponen yang meliputi:
   a. *use case diagrams*
   b. *sequence diagrams*
   c. *collaboration diagrams*
   d. *statechart diagrams*
   e. *activity diagrams*

Class diagrams merupakan gambar grafis mengenai struktur objek statis dari suatu sistem, menunjukkan *class-class* objek yang menyusun sebuah sistem dan juga hubungan antara kelas objek tersebut.

Object diagrams merupakan contoh model kelas dan digunakan untuk menggambarkan sistem pada titik waktu tertentu. Sama seperti objek yang merupakan contoh dari *class*, *object diagrams* sebagai sebuah contoh dari sebuah *class diagrams*.

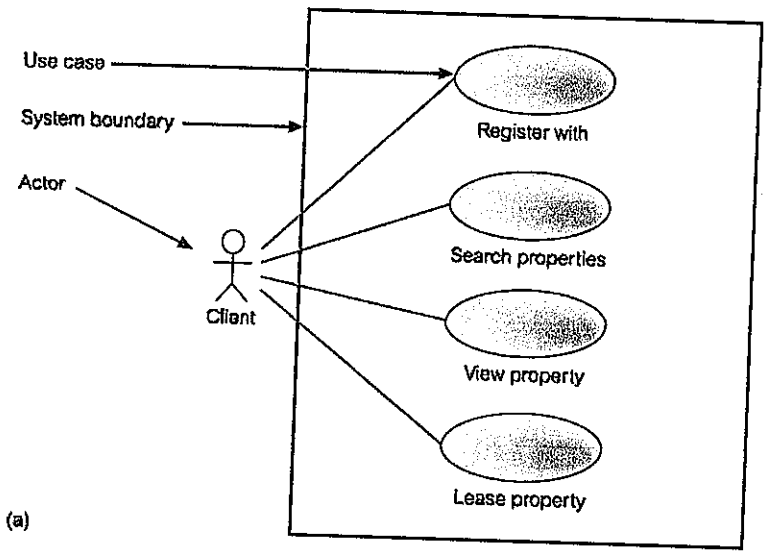


Gambar 2.4 *Object Diagrams* (Connolly&Begg, 2005, p838)

Component diagrams menggambarkan organisasi dan ketergantungan di antara komponen fisik perangkat lunak, seperti *source code*, *runtime code*, dan *executable*. Sebagai contoh, komponen diagram dapat menggambarkan ketergantungan antara file sumber dan file eksekusi.

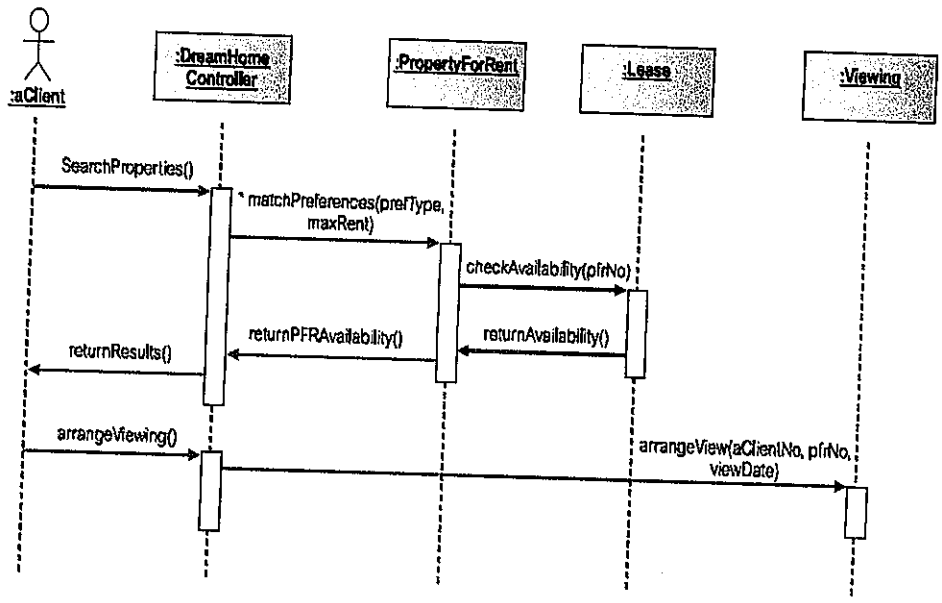
Deployment diagrams menggambarkan konfigurasi sistem *runtime*, menampilkan node perangkat keras, komponen yang berjalan pada node tersebut, dan hubungan antar node.

Use case diagrams menggambarkan model kasus dimana pengguna yang berinteraksi dengan sistem (aktor) dan hubungan antara pengguna dengan fungsi tersebut. Dengan kata lain, *use case* menggambarkan siapa yang akan menggunakan sistem dan dengan cara apa pengguna mengharapkan untuk berinteraksi dengan sistem.



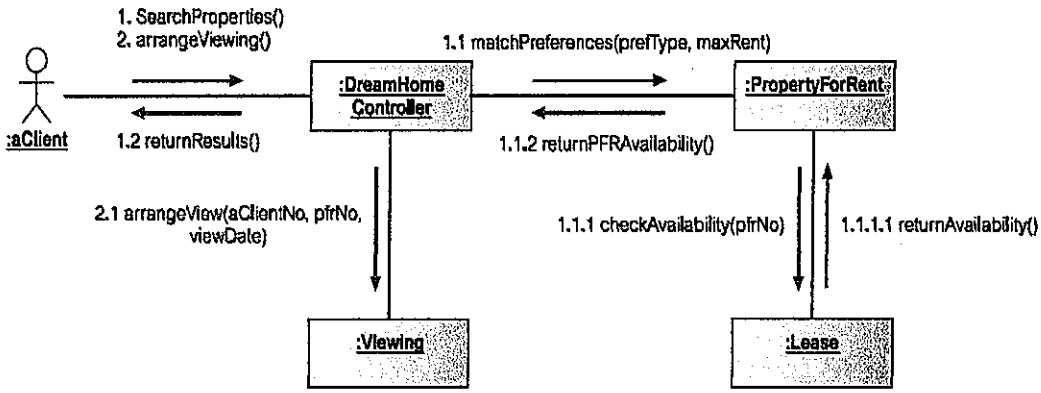
Gambar 2.5 *Use Case Diagrams* (Connolly&Begg, 2005, p840)

Sequence diagrams memodelkan sebuah *use case* dengan cara menggambarkan interaksi pesan di antara objek-objek dari waktu ke waktu.



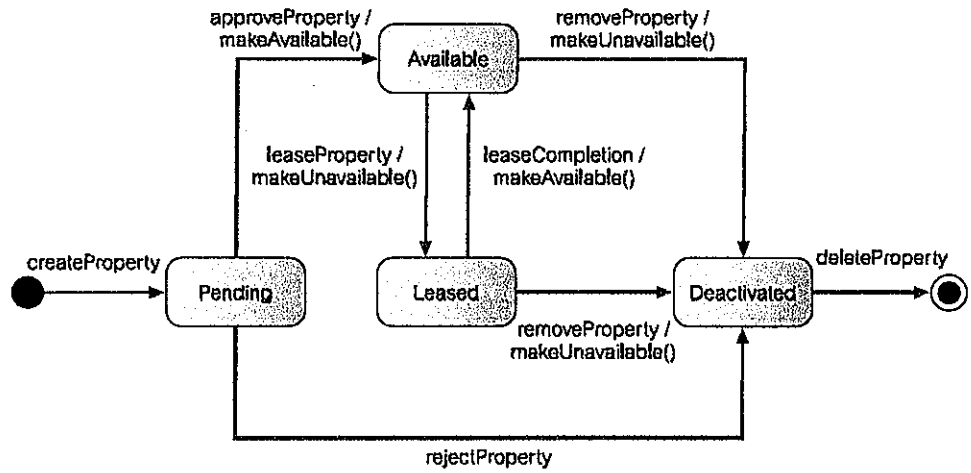
Gambar 2.6 *Sequence Diagrams* (Connolly&Begg, 2005, p841)

Collaboration diagrams adalah jenis lain dari diagram interaksi yang menggambarkan aliran pesan di antara objek-objek dalam rangkaian pesan.



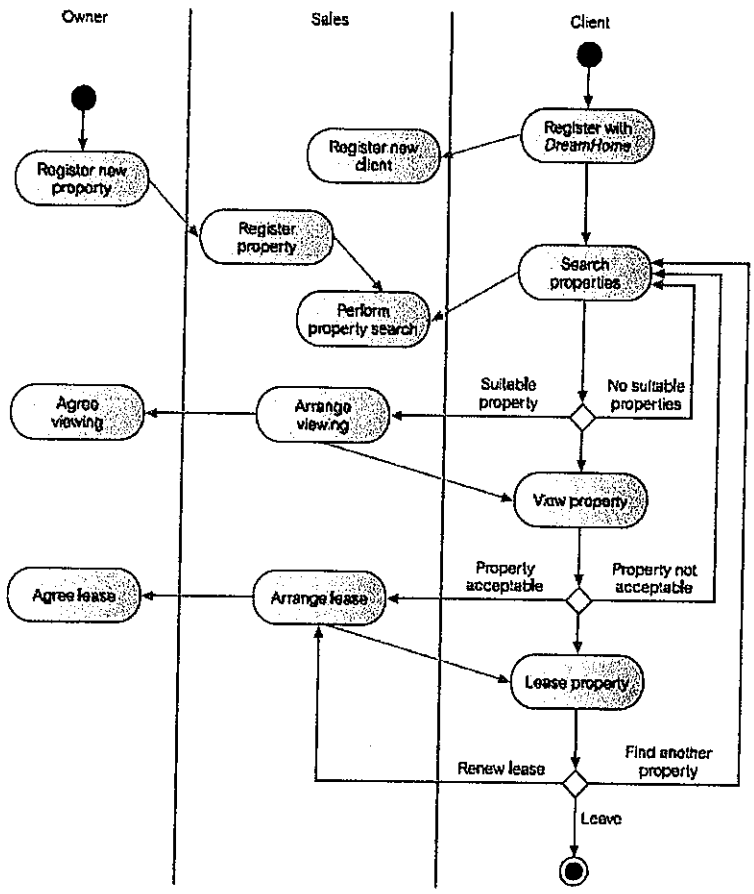
Gambar 2.7 *Collaboration Diagrams* (Connolly&Begg, 2005, p841)

Statechart diagrams disebut juga sebagai *state diagrams* yang menggambarkan bagaimana objek dapat berubah dalam menanggapi peristiwa eksternal. *Statechart* biasanya merupakan model transisi dari suatu objek tertentu.



Gambar 2.8 *Statechart Diagrams* (Connolly&Begg, 2005, p842)

Activity diagrams menggambarkan langkah-langkah *use case* atau sebuah langkah dalam suatu proses bisnis atau seluruh proses bisnis yang terdiri dari aktivitas *state* dan transisi.



Gambar 2.9 *Activity Diagrams* (Connolly&Begg, 2005, p843)

2.1.16 Perancangan Basis Data

Menurut Connolly dan Begg (2005, p291), perancangan basis data merupakan sebuah proses pembuatan sebuah rancangan bagi basis data yang akan mendukung misi dan tujuan perusahaan untuk sistem basis data yang diperlukan.

Tahap-tahap perancangan basis data dibagi menjadi tiga bagian, yaitu :

1. Perancangan basis data konseptual

   Menurut Connolly dan Begg (2005, p439), perancangan basis data konseptual adalah sebuah proses pembuatan model data yang digunakan dalam suatu perusahaan, yang terbebas dari semua pertimbangan fisikal seperti DBMS target, program aplikasi, bahasa pemrograman, *hardware* dan sebagainya. Perancangan basis data konseptual terbagi menjadi beberapa tahap yaitu :

   Langkah 1 : Membangun model data konseptual lokal bagi setiap *view*

   Bertujuan untuk membangun model data konseptual lokal dari sebuah perusahaan bagi setiap *view* tertentu. Selain analisa, sejumlah *user view* telah diidentifikasi dan tergantung pada jumlah *overlap* antara *view* tersebut. Beberapa *user view* mungkin telah dikombinasikan bersama untuk membentuk *view* bersama yang diberi nama yang sesuai. Tahapan-tahapan dari membangun model data konseptual diantaranya adalah :

   a. Mengidentifikasikan tipe entitas

      Tipe entitas dapat dikenali dengan mengidentifikasikan kata benda atau frase kata benda pada spesifikasi kebutuhan pengguna, objek besar seperti orang (*people*), tempat (*place*), benda (*thing*) atau konsep (*concept*) (Conolly dan

Begg, 2005, p443). Alternatif lain adalah dengan mencari obyek yang keberadaannya beda.

b. Mengidentifikasikan tipe relasi

Bertujuan untuk mengidentifikasi relasi yang penting yang ada antara tipe entitas-tipe entitas yang telah didefinisikan sebelumnya (Conolly dan Begg, 2005, p445). Tipe relasi diidentifikasikan dengan mencari kata kerja atau suatu kata yang berhubungan dengan kata kerja. Mengidentifikasikan dan mengasosiasikan atribut dengan entitas atau tipe relasi.

c. Mengidentifikasikan data menghubungkan atribut dengan entitas atau hubungan

Tujuannya untuk menghubungkan atribut dengan entitas dan tipe relasi yang tepat. Atribut yang dimiliki setiap entitas dan relasi harus memenuhi karakteristik atribut yaitu *simple/composite attributem single/multivalued attribute,* dan *derived attribute.*

d. Menentukan *domain* atribut

Domain adalah sekumpulan nilai dimana satu atau lebih atribut memperoleh nilainya (Conolly dan Begg, 2005, p450).

e. **Menentukan atribut-atribut *candidate key* dan *primary key***

Tujuannya untuk mengidentifikasikan *candidate key* dari setiap tipe entitas dan jika ada lebih dari 1 *candidate key.*

Primary key adalah *candidate key* yang dipilih secara unik mengidentifikasikan suatu tipe entitas. (Conolly dan Begg, 2005, p451).

Candidate key adalah kumpulan minimal dari atribut yang secara unik mengidentifikasi setiap tipe entitas. (Conolly dan Begg, 2005, p451).

f. **Mempertimbangkan penggunaan konsep *enhanced modeling* (optional)**

Tujuannya untuk mempertimbangkan penggunaan konsep *enhanced modeling* seperti spesialisasi/generalisasi, agregasi, dan komposisi.

g. **Memeriksa model untuk redudansi**

Tujuannya untuk memerikasa adanya redudansi didalam model.

Dua aktivitas yang ada di dalam tahapan ini adalah:

- **Memeriksa kembali *one-to-one (1:1) relationship***

  Kemungkinan ada dua entitas yang menggambarkan objek yang sama dalam

organisasi. Oleh karena itu, kedua entitas tersebut harus digabungkan.

- **Menghilangkan relasi yang redundan**

  Suatu relasi menjadi redundan jika informasi yang sama dihasilkan melalui relasi yang lainnya. Untuk meminimalkan data model maka relasi yang redundan harus dihilangkan.

h. Memvalidasi model konseptual dengan transaksi *user*

Tujuannya untuk memastikan model konseptual mendukung kebutuhan transaksi.

Ada dua pendekatan yang dapat memastikan bahwa model data konseptual mendukung kebutuhan transaksi:

- Menjelaskan transaksi tersebut

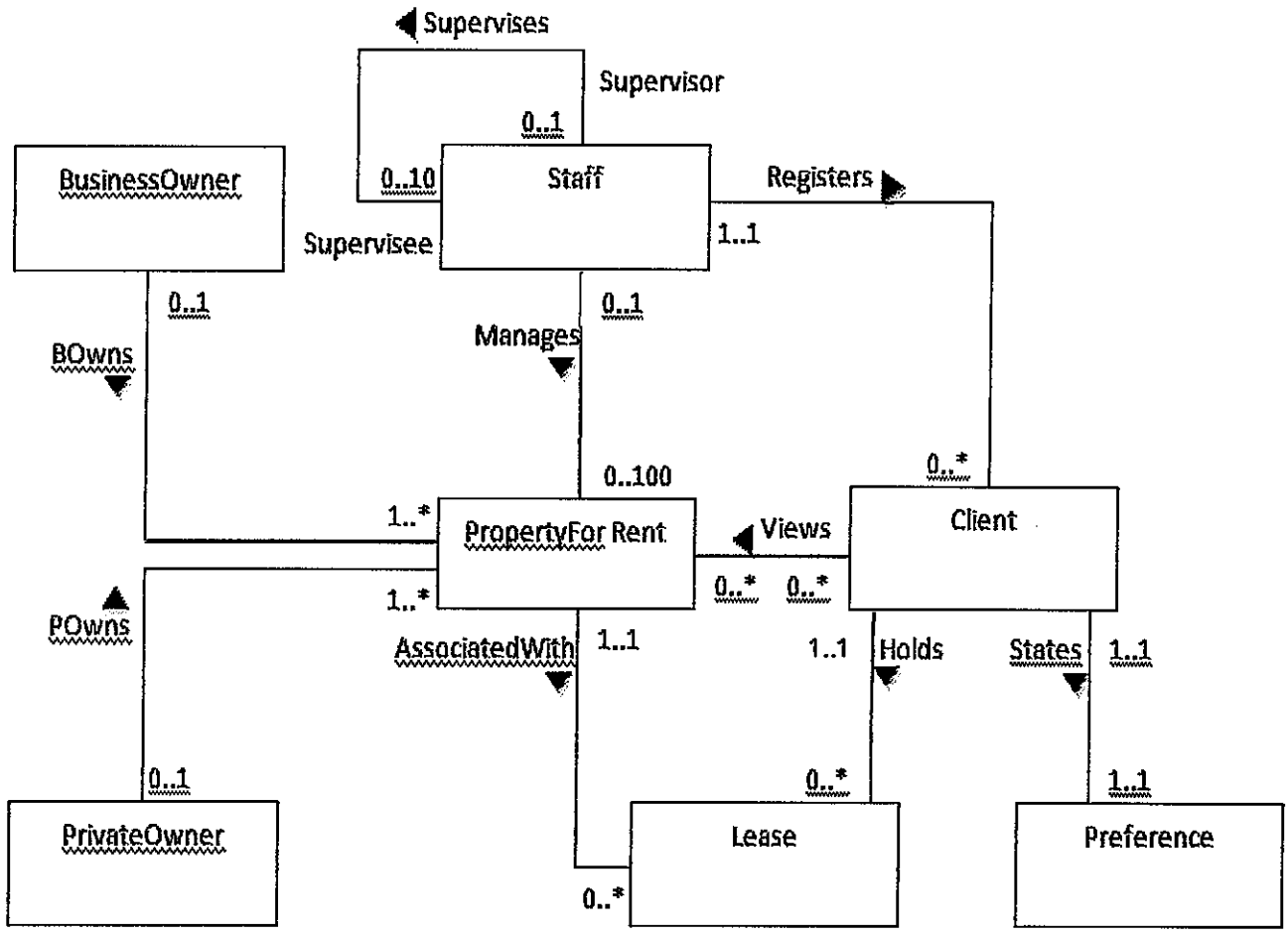
  Memeriksa semua informasi (entitas, relasi, dan atributnya) yang dibutuhkan setiap transaksi yang disediakan oleh model (Conolly dan Begg, 2005, p456).

- Menggunakan jalur transaksi

  Memvalidasi data model terhadap kebutuhan transaksidengan menggambar diagram yang mewakili *pathway* yang diambil oleh setiap transaksi secara langsung pada E-R diagram (Conolly dan Begg, 2005, p457).

i. Mengkaji ulang model data konseptual dengan *user*

Tujuannya untuk mengkaji ulang model data konseptual dengan *user* untuk memastikan bahwa model data konseptual merupakan representasi yang benar mereka akan mempertimbangkan model tersebut menjadi perwakilan yang sebenarnya dari kebutuhan data dalam perusahaan.



Gambar 2.10 Contoh ER Diagram Basis Data Konseptual (Connolly&Begg, 2005, p446)

2. Perancangan basis data logikal

Menurut Connolly dan Begg (2005, p439), perancangan basis data logikal merupakan sebuah proses pembuatan model dari informasi yang digunakan dalam suatu perusahaan berdasarkan data model spesifik, tetapi terbebas dari DBMS tertentu dan pertimbangan fisikal lainnya. Dalam bagian ini terdapat beberapa tahap yaitu :

Langkah 2 : Membangun dan memvalidasi model data logikal lokal bagi setiap *view*

Bertujuan untuk membangun model data logikal lokal dari model data konseptual lokal yang menggambarkan *view* tertentu dari sebuah perusahaan dan kemudian memvalidasi model tersebut untuk memastikan bahwa model terstruktur dengan baik dan mendukung transaksi yang diperlukan. Tahapan-tahapan dari membangun model data logical diantaranya adalah :

a. **Menghilangkan fitur-fitur yang tidak kompatibel dengan model relasional**

Bertujuan untuk menyaring *local conceptual data model* sehingga fitur-fitur yang tidak sesuai dengan model relasional dapat dihilangkan. Langkah-langkahnya antara lain:

- **Menghilangkan *many-to-many* (*:*) *binary relationship***

  Dengan memecah relasi yang mengandung *many-to-many* (*:*) untuk mengidentifikasikan sebuah entitas tengah (*intermediate entity*) sehingga relasi ini digantikan dengan dua buah *one-to-many* (1:*) *relationship*, dengan entitas tengan berada diantara dua buah entitas yang lama.

- **Menghilangkan *many-to-many* (*:*) *recursive relationship types***

  Jika *recursive relationship* ada pada *conceptual* data model, *relationship* tersebut harus dipecah untuk mengidentifikasikan sebuah entitas tengah dengan cara menganggap entitas yang terlibat pada relasi ini merupakan dua buah entitas dengan jenis relasi *many-to-many* (*:*) *binary* sehingga penyelesaiannya sama dengan penyelesaian pada relasi *many-to-many* (*:*) *binary*.

- **Menghilangkan *complex relationship types***

  Dihilangkan dengan memecah relasi ini untuk mengidentifikasikan entitas tengah (*intermedaite*

entity). Kemudian *complex relationship ini akan digantikan dengan beberapa one-to-many* (1:*) *binary relationship*

- **Menghilangkan multi-valued attributes**

  Cara menghilangkanya adalah dengan memecah atribut ini untuk mengidentifikasikan sebuah entitas.

b. Membuat relasi untuk model data logikal

Yang ditentukan pertama kali adalah nama relasi diikuti daftar *simple attribute* yang disertai dengan tanda kurung, *primary key* beserta *alternate key* dan atau *foreign key* dari *relasi.* Relasi antara satu entitas dengan entitas lainnya digambarkan dengan mekanisme *primary key* atau *foreign key.*

c. Memvalidasi hubungan dengan menggunakan normalisasi

Bertujuan untuk memvalidasi hubungan di dalam model data logical menggunakan normalisasi.

Ada beberapa tahapan dari normalisasi antara lain:

- *First Normal Form* (1NF)

  Sebuah relasi dikatakan 1NF jika titik temu tiap baris dan kolom pada relasi tersebut mengangandung satu dan hanya satu nilai

(Conolly, Begg, 2005, p403). Sebuah relasi akan berada dalam bentuk 1NF jika *repeating group*-nya sudah hilang. Ada dua pendekatan untuk menghilangkan *repeating group* pada table yang tidak normal, yaitu:

- Dengan memasukkan data yang sesuai ke dalam kolom yang kosong dari baris yang mengandung data berulang.
- Dengan menempatkan data yang berulang bersama salinan atribut kunci pada relasi yang terpisah. Sebuah *primary key* diidentifikasikan kedalam relasi yang baru.

- *Second Normal Form* (2NF)

  Relasi dikatakan 2NF jika relasi tersebut berada pada 1NF dan setiap atribut yang bukan *primary key* bergantung penuh (*fully functionally dependent*) terhadap *primary key* (Conolly dan Begg, 2005, p407)

  Full functionally dependency terjadi jika A dan B merupakan atribut dari suatu relasi, dan B dikatakan bergantung penuh terhadap A(A->B), jika B bergantung terhadap A, namun bukan subset dari A.

Untuk menghasilkan relasi dalam bentuk 2NF melibatkan penghilangan ketergantungan sebagian (*partial dependency*) dan menempatkan relasi yang baru bersama salinan atribut penentunya (*determinant attribute*).

- *Third Normal Form* (3NF)

  Suatu relasi dikatakan 3NF jika relasi tersebut berada dalam bentuk 1NF dan 2NF, dan tidak ada atribut bukan *primary key* bergantug secara transitif (*transitively dependent*) terhadap *primary key* (Conolly dan Begg, 2005, p408).

  Transitively dependency adalah sebuah kondisi dimana A,B, dan C merupakan atribut dari relasi yang jika A->B dan B->C mana Cdisebut bergantung secara transitif (*transitively dependent*) terhadap A melalui B (A tidak *functionally dependent* terhadap B atau C) (Conolly dan Begg, 2005, p408).

d. Memvalidasi hubungan dengan transaksi pengguna

Bertujuan untuk memastikan bahwa hubungan di dalam model data logikal mendukung kebutuhan transaksi (biasanya penggambaran dalam bentuk *view*).

e. **Memeriksa *integrity constraint***

- ***Required data***

  Beberapa atribut harus selalu berisi nilai yang benar (*valid*), tidak dapat bernilai *null*. *Constraint* ini harus diidentifikasikan pada saat mendokumentasikan atribut-atribut pada kamus data (langkah 1.3).

- ***Attribute domain constraint***

  Setiap atribut memiliki domain, yaitu himpunan nilai yang diperbolehkan (Conolly dan Begg, 2005, p475). *Constraint* ini harus diidentifikasikan pada saat mendokumentasikan atribut-atribut pada kamus data (langkah 1.4).

- ***Entity integrity***

  Primary key dari sebuah entitas tidak boleh bernilai *null*. *Constraint* ini harus dipertimbangkan pada saat penentuan *primary key* bagi setiap tipe entitas (langkah 1.5).

- ***Referential integrity***

  Jika suatu *foreign key* memiliki nilai, maka niali tersebut harus menunjuk kesebuah baris yang ada pada relasi *'parent'*.

- ***General constraint (business rules)***

Kegiatan *update* entitas dibatasi oleh peraturan atau kebijakan organisasi yang mengatur transaksi yang diwakilkan oleh *update* yang dilakukan.

f. Mengkaji ulang model data logikal dengan pengguna

Bertujuan untuk meninjau ulang model data logikal dengan *user* untuk memastikan bahwa mereka mempertimbangkan model tersebut untuk menjadi representasi nyata dari kebutuhan data di dalam sebuah perusahaan.

g. Menggabungkan data model logikal menjadi model global (optional)

Bertujuan menggabungkan masing-masing lokal logikal data model menjadi sebuah global logikal data model yang menggambarkan organisasi dengan menyatukan masing-masing lokal logikal data model bagi setiap pandangan pengguna.

h. Memeriksa pertumbuhan lebih lanjut

Bertujuan untuk menentukan apakah ada perubahan yang signifikan untuk masa depan yang sudah dapat diduga sebelumnya dan menilai apakah model data logikal dapat mengakomodasi perubahan ini.

Langkah 3 : Membangun dan memvalidasi model data logikal global

Bertujuan untuk mengkombinasikan model data logikal lokal individu ke dalam model data logikal global tunggal yang menggambarkan perusahaan. Kemudian memvalidasi model global, pertama terhadap aturan normalisai, dan yang kedua terhadap transaksi yang dijelaskan dalam semua *view*.

a. Mendesign relasi dasar

Tujuan dari langkah ini adalah untuk memutuskan bagaimana ikan relasi dasar yang diidentifikasikan di dalam model data logikal ke dalam target DBMS.

Untuk setiap relasi yang diidentifikasikan pada model data logikal global, definisinya terdiri dari :

- Nama Relasi
- Suatu *list* untuk atribut yang sederhana
- *Primar key, alternate key,* dan *foreign key*
- Suatu daftar dari atribut turunan dan bagaimana pembuatannya
- Batasan integrasi untuk setiap *foreign key* yang diidentifikasi.

Dari kamus data, dari setiap atributnya dapat diketahui:

- *Domain* atribut tersebut, yang terdiri dari tipe data, panjang, dan berbagai batasan dalam domain.

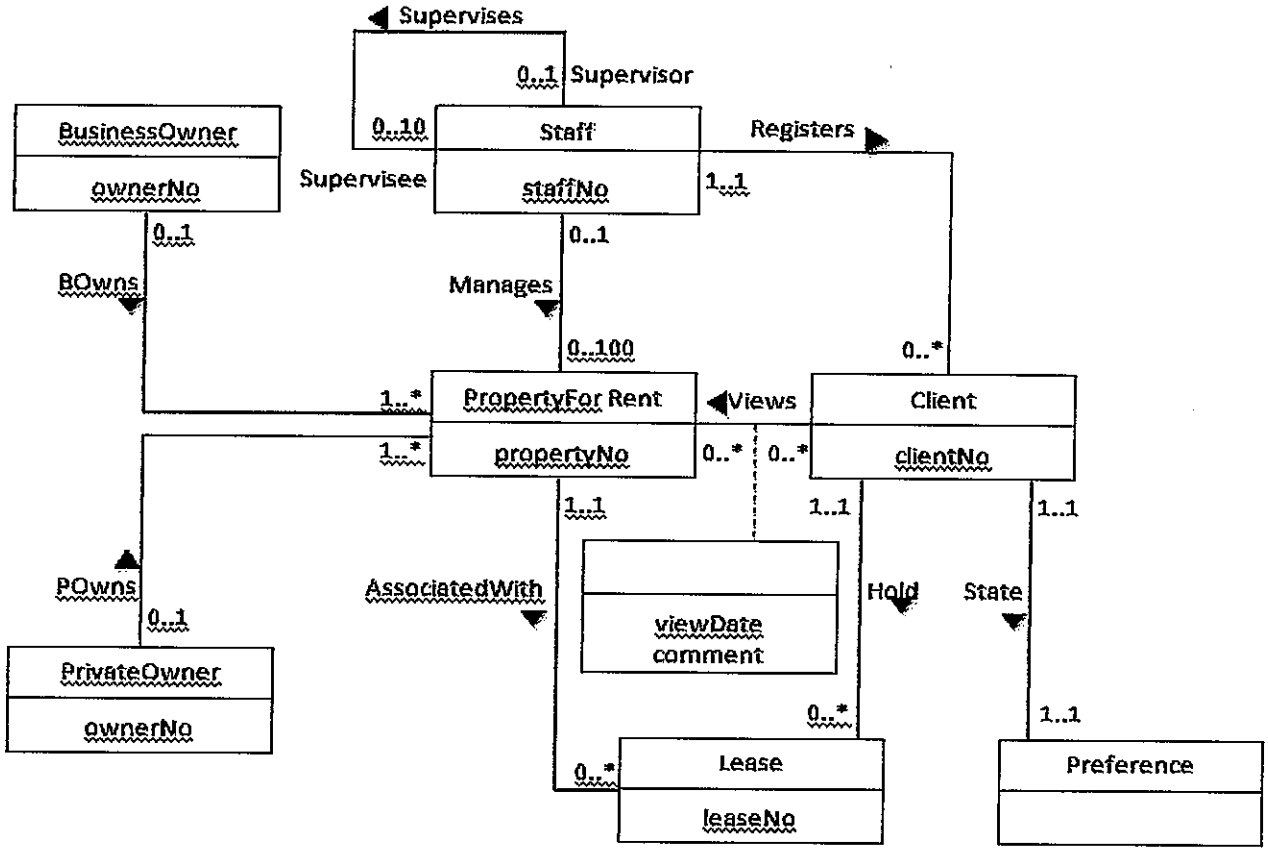
- Sebuah *optional* nilai *default* untuk atribut.
- Atribut boleh bernilai *null*.
- Atribut diperoleh dan bagaimana atribut tersebut dikomputerisasi.

b. Merancang representasi data dari turunan

Bertujuan untuk memutuskan bagaimana untuk mempresentasikan berbagai data turunan pada model data logikal di dalam DBMS.

c. Merancang batasan *general*

Bertujuan untuk merancang batasan *general* untuk DBMS yang digunakan.



Gambar 2.11 Contoh ER Diagram Basis Data Logikal (Connolly&Begg, 2005, p452)

3. Perancangan basis data fisikal

Menurut Connolly dan Begg (2005, p439), perancangan basis data fisikal adalah sebuah proses pembuatan deskripsi implementasi basis data pada tempat penyimpanan kedua. Perancangan ini menjelaskan relasi dasar, organisasi file, dan index yang digunakan untuk mencapai akses yang efisien ke data dan berbagai batasan integritas yang berhubungan serta penilaian keamanan. Di dalam perancangan ini terdapat beberapa tahap, yaitu:

Langkah 4 : Menterjemahkan model data logikal global bagi DBMS target

Bertujuan untuk menghasilkan skema basis data relasional dari model data logikal global yang dapat diimplementasikan dalam DBMS target. Bagian pertama dari proses ini adalah mengumpulkan informasi yang diperoleh selama perancangan basis data logikal dan terdokumentasi dalam kamus data. Bagian kedua adalah menggunakan informasi ini untuk membuat rancangan relasi dasar. Proses ini memerlukan pengetahuan yang mendalam dari fungsi yang ditawarkan oleh DBMS target.

Langkah 5 : Merancang gambaran fisikal

Bertujuan untuk menentukan organisasi file yang optimal untuk menyimpan relasi dasar dan index yang diperlukan untuk mencapai performa yang diinginkan, yaitu sebuah cara dimana relasi dan *tuple* akan disimpan pada tempat penyimpanan kedua.

Tujuan utama dari perancangan basis data fisikal adalah untuk menyimpan data dalam cara yang efisien.

Langkah 6 : Kebutuhan ruang *hardisk*

Bertujuan umtuk memperkirakan jumlah ruang *hardisk* yang diperlukan oleh DBMS. Hal ini tergantung pada DBMS target dan *hardware* yang digunakan untuk mendukung basis data. Pengukuran didasarkan pada ukuran tiap *tuple* dan jumlah *tuple* dalam relasi.

Langkah 7 : Merancang mekanisme keamanan

Bertujuan untuk merancang mekanisme keamanan bagi basis data yang ditentukan oleh pengguna. Basis data menggambarkan sumberdaya perusahaan yang penting sehingga keamanan dari sumberdaya tersebut sangatlah penting. Selama tahap pengumpulan dan analisa kebutuhan, kebutuhan keamanan tertentu harus sudah didokumentasikan di dalam spesifikasi kebutuhan sistem. Tujuan dari tahap ini adalah merealisasikan kebutuhan keamanan yang telah ditentukan.

2.1.17 Normalisasi

Menurut Whitten, Bentley dan Dittman (2004, p306), normalisasi didefinisikan sebagai suatu teknik analisis data yang mengelola data ke dalam kelompok-kelompok untuk membentuk entitas yang nonredundan, adaptif dan fleksibel.

Pengertian normalisasi menurut Connolly dan Begg (2005, p387) adalah suatu teknik untuk memproduksi satu set hubungan dengan kebutuhan yang diinginkan, memberi kebutuhan data dari suatu perusahaan. Tujuan lain dari normalisasi adalah mencegah adanya redundansi atau pengulangan data yang nantinya akan menghemat tempat pada penyimpanan. Menurut Connolly dan Begg, proses dalam normalisasi terbagai dalam beberapa tahap, yaitu :

- *Unnormalized Form (UNF)*

  Suatu tabel dikatakan sebagai bentuk *unnormalized* bila didalamnya terdapat kelompok berulang atau yang biasa dikenal *repeating group.*

- *First Normal Form (*1NF)

  Untuk mengubah bentuk *Unnormalized Form* menjadi 1NF, yang harus dilakukan adalah mengidentifikasi dan menghilangkan kelompok berulang, agar setiap pertemuan antara baris dan kolom berisi satu dan hanya satu nilai. Dilihat dari contoh bentuk *unnormalized* sebelumnya, dapat diidentifikasi kelompok berulang sebagai berikut: Hari, Waktu, KdMtk, MataKuliah, SKS, Ruang, dan Kelas.

  Langkah-langkah membuat normal pertama dari bentuk unnormal:

  1. Menentukan satu atau lebih atribut sebagai atribut kunci dari tabel *unnormal*

2. Mengidentifikasi *repeating group* dari suatu tabel *unnormal* yang mengulang atribut kunci diatas.
3. Menghilangkan *repeating group,* dapat dilakukan dengan 2 cara : mengisi kolom dan baris yang kosong dengan data-data yang sesuai atau menempatkan data yang berulang beserta *key* nya kedalam tabel baru.

- *Second Normal Form* (2NF)

Pada tahap normalisasi 2NF dihilangkan setiap *Partial Dependence* yang ada pada bentuk 1NF. Yang dimaksud dengan *Partial Dependence* adalah atribut *non primary key* yang merupakan sebagian fungsi dari *primary key*, atau dapat dijelaskan demikian apabila terdapat atribut-atribut dalam suatu relasi yang memiliki ketergantungan fungsional misalnya atribut A (Kd_dosen, Nama) dan atribut B (KdMtk), dikatakan *partial dependence* apabila ada sebagian atribut dari A dihilangkan namun ketergantungan masih ada. Langkah-langkah membuat 2NF dari 1NF :

1. Mengidentifikasi *primary key* dari bentuk 1NF.
2. Mengidentifikasi *functional dependency* pada bentuk 1NF.
3. Apabila terdapat *partial dependency* dalam *primary key* maka tempatkan *primary key* tersebut dalam suatu tabel baru beserta field-field yang berkaitan dengannya.

- *Third Normal Form* (3NF)

Pengujian terhadap bentuk normal ketiga dilakukan dengan cara melihat apakah terdapat atribut bukan *key* tergantung fungsional terhadap atribut bukan *key* yang lain (disebut ketergantungan *transitive*). Pada tahap normalisasi 3NF dihilangkan setiap *Transitive Dependence* yang terdapat pada bentuk 2NF. Kondisi *transitive dependence* dapat diterangkan sebagai berikut : misalkan terdapat atribut A, B, dan C yang mempunyai relasi A→B, dan B→C, C adalah *transitive dependence* terhadap A melalui B.

Ketergantungan transitif (*transitive dependency*) terjadi ketika ada atribut yang bukan merupakan *primary key*, yang memiliki ketergantungan pada atribut lain yang juga bukan merupakan *primary key*. Langkah-langkah membuat 3NF dari 2NF :

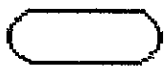
1. Mengidentifikasi *primary key* pada bentuk 2NF.
2. Mengidentifikasi *functional dependency* dalam tabel tersebut.
3. Apabila terdapat *transitive dependency* pada *primary key* maka tempatkan *primary key* beserta field-field yang berkaitan pada tabel baru.

Proses normalisai secara umum dilakukan sampai tahap 3NF karena sudah tidak terdapat data pengulangan, dan anomaly yang ada sudah sangat sedikit.

2.1.18 Flowchart

Flowchart adalah representasi grafik dari langkah-langkah yang harus diikuti dalam menyelesaikan suatu permasalahan yang terdiri atas sekumpulan simbol, dimana masing-masing simbol merepresentasikan suatu kegiatan tertentu.

Berikut adalah beberapa simbol yang digunakan dalam menggambar suatu flowchart :

SIMBOL	NAMA	FUNGSI
	TERMINATOR	Permulaan/akhir program
	GARIS ALIR (FLOW LINE)	Arah aliran program
	PREPARATION	Proses inisialisasi/pemberian harga awal
	PROSES	Proses perhitungan/proses pengolahan data
	INPUT/OUTPUT DATA	Proses input/ output data, parameter informasi
	PREDEFINED PROCESS (SUB PROGRAM)	Permulaan sub program/proses menjalankan sub program
	DECISION	Perbandingan pernyataan, penyeleksian data yang memberikan pilihan untuk langkah selanjutnya
	ON PAGE CONNECTOR	Penghubung bagian-bagian flowchart yang berada pada satu halaman
	OFF PAGE CONNECTOR	Penghubung bagian-bagian flowchart yang berada pada halaman berbeda

Tabel 2.1 Simbol-simbol Flowchart

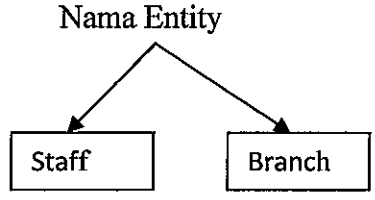
2.1.19 Entity Relationship Diagram (ERD)

Menurut Connolly dan Begg (2005, p342) ERD adalah suatu pendekatan secara top-down untuk didesain basis data dengan cara mengidentifikasi entity dan relasi antar data yang harus direpresentasikan dalam model.

Menurut Silberschatz (2005, p42) ERD adalah diagram yang digunakan untuk mengekspresikan struktur atau skema logikal dari basis data dalam bentuk grafis.

2.1.19.1 Tipe entiti

Tipe entiti adalah sekumpulan dari objek dengan sifat *properties* yang sama, dan keberadaannya tidak tergantung (*independent existence*). Sebuah tipe entity memiliki keberadaan yang bebas dan bisa menjadi objek dengan keberadaan fisik (*real*) atau menjadi objek dengan keberadaan konseptual (*abstrak*). Artinya perancangan yang berbeda mungkin mengidentifikasi entity yang berbeda pula. Sebuah basis data biasanya berisi banyak tipe entity yang berbeda. Dalam UML, huruf pertama dari nama entity diawali dengan huruf kapital. *Entity occurrence* adalah sebuah objek unik yang diidentifikasi dari sebuah tipe entiti.



Gambar 2.12 Contoh Tipe Entiti

2.1.19.2 Relationship

Relationship adalah hubungan yang berarti antara tipe entity yang satu dengan yang lain, dimana tiap *relationship* diberi nama yang menggambarkan fungsi yang berbeda.

2.1.19.3 Attribute dan Key

Attribut adalah sifat dari sebuah entiti atau tipe *relationship*. Sifat tertentu inilah yang disebut sebagai atribut. Atribut menyimpan nilai dari setiap *entity occurrence* dan merupakan bagian utama dari data yang disimpan dalam basis data.

Atribut *domain* adalah sejumlah nilai yang diperkenankan oleh satu atau lebih atribut. Setiap atribut yang dihubungkan dengan sejumlah nilai disebut *domain.*

Simple attribute (*atomic attribute*) adalah sebuah susunan atribut dari komponen tunggal yang keberadaannya bebas.

Single value attribute merupakan atribut yang hanya menyimpan nilai tunggal untuk suatu sifat dari entiti. *Multi-valued attribute* merupakan atribut yang bisa menyimpan nilai lebih dari satu untuk suatu sifat dari entiti.

Derived attribute adalah nilai suatu atribut diperoleh dari atribut yang saling berhubungan atau kumpulan atribut yang saling berhubungan.

Candidate key merupakan sejumlah kecil atribut yang secara unik mengenali setiap kejadian didalam tipe entiti. Sebuah *candidate key* tidak boleh bernilai null.

Primary key merupakan *candidate key* yang digunakan untuk mengenali secara unik setiap *occurrence* dari sebuah tipe entiti. Kepemilikan *primary key* berdasarkan pada panjang atribut, jumlah minimal dari kebutuhan atribut dan keunikannya. *Candidate key* yang tidak terpilih disebut sebagai *alternate key.*

Foreign key adalah atribut pada satu relasi yang cocok pada *candidate key* dari beberapa relasi.

2.1.19.4 Constraint dan structure

Constraint harus mencerminkan pembatasan pada *relationship* perkalian (*perceived*) didalam kenyataan. Tipe utama dari *constraint* didalam *relationship* disebut *multiplicity.*

Multiplicity adalah jumlah kemungkinan *occurrence* dari sebuah entiti yang bisa menghubungkan suatu *occurrence* tunggal dari hubungan entiti melalui *relationship* tertentu.

Multiplicity mengandung dua *constraint* yang dipisahkan yang dikenal sebagai *cardinality* dan *participation. Cardinality* menggambarkan jumlah maksimum kemungkinan *relationship occurrence* untuk sebuah entiti *participation* dalam *relationship* yang diberikan.

Participation menetapkan semua atau beberapa *occurrence* yang diikutsertakan dalam sebuah tipe *relationship.*

Sebuah *participation constraint* yang menunjukkan semua entiti *occurrence* terkait dalam *relationship* tertentu disebut sebagai *mandatory participation,* sedangkan yang hanya menunjukkan beberapa disebut *optional participation.*

Cara alternative menggambarkan *Multiplicity constraint*	Arti
0...1	Nol atau satu entiti *occurrence*
1...1 (atau 1)	Hanya satu entiti *occurrence*
0...*(atau *)	Nol atau banyak entiti *occurrence*
1...*	Satu atau banyak entiti *occurrence*
5...10	Minimum 5 dan maksimum 10 entiti *occurrence*
0,3,6-8	Nol atau tiga atau enam, tujuh atau delapan entiti *occurence*

Tabel 2.2 Cara alternative menggambarkan *Multiplicity Constraint*

2.1.19.5 Integrity constraint

Integrity constraint merupakan batasan yang kita inginkan untuk mencegah basis data tidak konsisten. *Integrity constraint* dibagi menjadi empat bentuk dasar, yaitu:

1. *Require Data,* beberapa atribut harus selalu mengandung nilai yang *valid,* dengan kata lain tidak boleh mengandung nilai null
2. *Atribut Domain Constraint,* setiap atribut mempunyai domain sendiri yaitu sekumpulan nilai yang sah untuk suatu atribut
3. *Entity Integrity, primary key* dari sebuah entiti tidak boleh mengandung nilai null\
4. *Referential Integrity,* sebuah *foreign key* menghubungkan setiap baris di dalam relasi anak kepada baris di dalam relasi induk yang mengandung nilai *candidate key* yang cocok.

2.1.20 PHP

PHP digagaskan pertama kali pada tahun 1994 oleh Rasmus Lerdorf dan telah melalui tiga fase penulisan ulang sampai yang saat ini digunakan. Menurut Welling dan Thomson (2001, p2), PHP adalah sebuah "*Server Side Scripting Language*" yang dirancang khusus untuk web. Dalam sebuah halaman HTML Anda dapat menanamkan kode PHP yang akan dijalankan setiap kali halaman dikunjungi. PHP merupakan produk *open source*.

2.1.21 SQL (*structure query language*)

SQL (*structure query language*) adalah bahasa basis data relasional yang paling populer dan yang memungkinkan para pengguna untuk melakukan berbagai pencarian rumit. SQL menggabungkan baik fitur DML maupun DDL. SQL menawarkan kemampuan untuk melakukan berbagai pencarian rumit dengan pernyataan yang sederhana. Kata kunci seperti SELECT (untuk menspesifikasikan atribut yang diinginkan), FROM (untuk menspesifikasi tabel yang akan digunakan), dan WHERE (untuk menspesifikasikan kondisi yang diterapkan dalam permintaan) biasanya digunakan untuk manipulasi data.

2.1.22 MySql

Menurut Welling dan Thomson (2001, p3), MySql telah tersedia untuk publik sejak tahun 1996. Mysql sekarang tersedia di bawah lisensi open source, namun lisensi komersial juga tersedia jika diperlukan. MySql sangat cepat, kuat, dbms relasional. Basis data memungkinkan Anda untuk secara efisien menyimpan, mencari, mengurutkan dan mengambil data. MySql Server mengontrol akses ke data Anda untuk memastikan bahwa beberapa pengguna dapat bekerja dengan itu concurently, untuk menyediakan akses cepat ke sana, dan memastikan bahwa hanya pengguna yang berwenang dapat memperoleh akses. Mysql adalah multi-user, multi-threaded server.

2.1.23 Perancangan Aplikasi

Menurut Connolly dan Begg (2005, p287), rancangan apliksi yaitu perancangan antarmuka pengguna dan program aplikasi yang digunakan dan memproses basis data. Aktivitas antara *database design* dengan *application design* terjadi secara paralel. Dalam banyak hal, tidak mungkin untuk menyelesaikan perancangan aplikasi sampai perancangan dari basis data itu sendiri. Dalam hal ini, basis data hadir untuk mendukung aplikasi dan harus ada aliran informasi antara *application design* dan *database design.*

2.1.24 Implementasi

Menurut Connolly dan Begg (2005, p292), implementasi adalah realisasi fisik dari rancangan aplikasi dan basis data. Implementasi basis data dicapai dengan menggunakan DDL dari DBMS yang dipilih atau GUI yang menyediakan fungsi yang sama selama penyembunyian *statement* DDL tingkat rendah. DDL digunakan untuk membuat struktur basis data dan file basis data kosong.

Program aplikasi diterapkan menggunakan bahasa tingkat tiga atau empat. Bagian dari program aplikasi ini adalah transaksi basis data, yang diterapkan menggunakan DML yang dapat dijalankan pada sekumpulan bahasa pemrograman, seperti *Visual Basic, C, C++, Java, COBOL,* atau *Pascal.*

2.1.25 Pengubahan dan Peng-*load*-an Data

Menurut Connolly dan Begg (2005, p292), pengubahan dan peng-*load*-an data adalah pentransferan berbagai data yang ada ke dalam basis data yang baru dan pengubahan berbagai aplikasi yang ada untuk berjalan pada basis data yang baru. Tahapan ini dibutuhkan hanya ketika sistem basis data yang baru menggantikan sistem yang lama. Sekarang ini, sudah menjadi hal yang biasa bagi sebuah DBMS untuk mempunyai utilitas yang dapat memuat keseluruhan file yang ada ke dalam basis data yang baru. Utilitas biasanya membutuhkan spesifikasi dari sumber dan tujuannya, sehingga mengubah data sesuai dengan format basis data yang baru.

2.1.26 Pengujian

Menurut Connolly dan Begg (2005, p293), pengujian adalah sebuah proses pengeksekusian program aplikasi dengan maksud menemukan kesalahan. Hal ini dicapai dengan strategi pengujian terencana yang hati-hati dan data yang nyata sehingga proses pengujian seluruhnya ditangani dengan metodologis dan benar. Jika proses pengujian berjalan dengan baik, hal ini akan menemukan banyak kesalahan dalam program aplikasi dan struktur basis data. Pengujian juga menunjukkan agar aplikasi basis data bekerja sesuai dengan spesifikasi dan kebutuhan performa yang diinginkan. Hasil dari pengujian dapat memberikan penilaian terhadap kehandalan dan kualitas *software*.

2.1.27 Pemeliharaan Operasional

Menurut Connolly dan Begg (2005, p293), pemeliharaan operasional adalah sebuah proses pemantauan dan pemeliharaan sistem berikut instalasi. Dalam tahap ini melibatkan dua aktivitas, yaitu :

- Pemantauan performa sistem : jika performa berada jauh dibawah level yang diharapkan, diperlukan perbaikan atau penyusunan kembali basis data.
- Pemeliharaan dan peng-*update*-an aplikasi basis data : kebutuhan baru dimasukkan ke dalam aplikasi basis data melalu tahap-tahao siklus hidup aplikasi basis data yang sebelumnya.

2.2 Teori-teori Khusus yang Berhubungan dengan Topik yang Dibahas

2.2.1 Hotel

Kata hotel mulai digunakan sejak abad ke 18 di London (Inggris), sebagai Hotel Garni yaitu sebuah rumah besar yang dilengkapi dengan sarana tempat menginap/ tinggal untuk penyewaan secara harian, mingguan atau bulanan.

Kata hotel sendiri merupakan perkembangan dari bahasa Perancis yaitu *Hostel*, yang diambil dari bahasa latin *Hospes*, dan mulai diperkenalkan kepada masyarakat umum pada tahun 1797. Sebelum istilah hotel digunakan di Inggris, rumah-rumah penginapan bagi orang yang bepergian jauh disebut *Inn*.

Menurut SK. Menteri perhubungan No. PM. 10/ Pw. 301/Phb.77, hotel didefinisikan sebagai suatu bentuk akomodasi yang dikelola secara

komersial, disediakan bagi setiap orang untuk memperoleh pelayanan dan penginapan berikut makan dan minum. Selain itu, hotel juga dapat didefinisikan sebagai berikut :

- Hotel adalah suatu usaha akomodasi komersial.
- Hotel harus dibuka untuk umum.
- Hotel harus memiliki suatu sistem pelayanan (*service system*).
- Hotel harus memiliki minimum 3 (tiga) macam fasilitas/ produk.
- Sedangkan definisi hotel menurut SK Menparpostel No.KM 34/HK 103/MPPT-87 adalah sebagai berikut:

  "Hotel adalah suatu jenis akomodasi yang mempergunakan sebagian atau seluruh bangunan untuk menyediakan jasa pelayanan penginapan, makan dan minum serta jasa lainnya bagi umum, yang dikelola secara komersial serta memenuhi ketentuan persyaratan yang ditetapkan dalam keputusan pemerintah".

2.2.2 Fungsi Hotel

Hotel memiliki fungsi antara lain sebagai berikut :

a. Sebagai tempat/ sarana akomodasi untuk memenuhi kebutuhan tamu (wisatawan dan pelancong), sebagai tempat beristirahat/ tinggal sementara waktu selama dalam

perjalanan yang jauh dari tempat asalnya. Oleh karena itu dalam bahasa Inggris hotel sering disebut sebagai "*Hotel is a home far away from home*".

b. Sebagai tempat pertemuan (rapat, seminar, konferensi, lokakarya, dan sebagainya) bagi para usahawan, pimpinan pemerintahan, para cendekiawan, dan sebagainya.

c. Sebagai tempat untuk mempromosikan berbagai produk, perusahaan, atau bisnis apa saja.

d. Sebagai tempat untuk bersantai, rekreasi, rileks, atau menikmati kesenangan lainnya.

e. Sebagai tempat untuk menambah ilmu pengetahuan dan pengalaman (khususnya bagi pelajar/ mahasiswa, dan karyawan).

f. Sebagai tempat untuk mencari nafkah/ uang (khususnya bagi karyawan dan *management*nya).

2.2.3 Peranan Hotel

Dalam menunjang pembangunan suatu negara, usaha Perhotelan dapat berperan aktif dalam berbagai hal, antara lain :

1. Meningkatkan peranan industri rakyat. Peranan ini dapat diwujudkan dengan adanya kebutuhan hotel akan peralatan dan perlengkapan untuk mendukung usaha pelayanan kepada tamu, antara lain : meubel, bahan-bahan makanan & minuman, bahan

pakaian, mesin-mesin, cindera mata, alat-alat kebersihan, hiasan bunga, dan lain-lain dihasilkan oleh industri rakyat.

2. Menciptakan lapangan kerja baru. Bisnis hotel merupakan usaha padat modal dan padat karya, yaitu memerlukan modal yang besar dengan jumlah tenaga kerja yang besar pula. Tenaga kerja dari berbagai latar belakang pendidikan dan pengalaman yang dapat dipekerjakan dalam berbagai bagian dan jabatan dalam hotel.
3. Membantu Pemerintah dan swasta dalam usaha pendidikan dan pelatihan. Hotel-hotel memberikan peluang yang sangat luas, kepada karyawan baru maupun lama untuk meningkatkan pengetahuan dan keterampilannya, dan juga memberikan kesempatan kepada para siswa/ mahasiswa untuk berpraktek kerja, magang kerja, bahkan langsung bekerja. Hotel merupakan tempat yang sangat efektif dalam pengembangan program alih teknologi (khususnya dalam usaha perhotelan) melalui penyerapan ilmu dan teknologi serta keterampilan kerja.
4. Meningkatkan pendapatan Pemerintah Daerah/ Negara dalam sector pajak. Sebagai usaha akomodasi dan jasa pelayanan, hotel merupakan salah satu sumber pendapatan daerah/ negara yang potensial, yaitu melalui pembayaran izin-izin, (Izin Perintis dan Izin Usaha Tetap, Izin Pengembangan Usaha), berbagai macam pajak, pembayaran listrik, telepon, dan berbagai macam izin/ pajak lainnya, baik kepada pemerintah daerah maupun kepada pemerintah pusat.

5. Meningkatkan devisa/ pendapatan Negara (dari sector pajak dan bea cukai). Hotel sebagai salah satu komponen indsutri pariwisata yang sangat berperan aktif dalam membantu meningkatkan arus wisatawan mancanegara, maupun wisatawan Nusantara, yang tentunya akan mempergunakan fasilitas dan pelayanan seperti : *passport, visa, exit permit* dan *entry permit*, tiket pesawat udara dan laut, dan lain-lain.

2.2.4 Kriteria Klasifikasi Hotel

Kriteria klasifikasi hotel di Indonesia secara resmi dikeluarkan oleh peraturan pemerintah, dalam hal ini dibawah Deparpostel dan dibuat oleh Dirjen Pariwisata dengan SK : Kep-22/U/VI/78.

Untuk mengklasifikasikan sebuah hotel dapat ditinjau dari berbagai faktor yang satu sama lain ada kaitannya. Faktor-faktor pengklasifikasian hotel tersebut, antara lain faktor tingkatan atau bintang dari hotel, faktor tujuan pemakaian, faktor lokasi hotel, faktor daya jual dan perencanaan penggunaan (*Hotel Plan Usage)*, faktor jumlah kamarnya, faktor ukuran hotel, faktor lamanya tamu menginap, faktor kegiatan tamu selama menginap, dan faktor jenis tamu yang menginap. Adapun keterangan dari faktor-faktor tersebut diatas adalah sebagai berikut :

2.2.4.1 Faktor tingkatan atau bintang dari hotel

Tingkatan atau kelas hotel dibedakan atas tanda bintang (*). Semakin banyak jumlah bintang maka persyaratan, fasilitas dan pelayanan (*service)* yang dituntut semakin banyak dan baik.

Kriteria klasifikasi hotel berdasarkan bintang adalah sebagai berikut :

1. Hotel berbintang satu (*).
2. Hotel berbintang dua (**).
3. Hotel berbintang tiga (***).
4. Hotel berbintang empat (****).
5. Hotel berbintang lima (*****).

Khusus untuk hotel berbintang V mempunyai tingkatan, yaitu *Palm, Bronze* dan *Diamond.*

2.2.4.2 Faktor tujuan pemakaian hotel selama menginap

Klasifikasi hotel berdasarkan faktor tujuan pemakaian selama menginap, dibagi menjadi dua bagian, yaitu :

1. *Business hotel*

   Hotel yang banyak digunakan oleh para usahawan. Hotel ini memiliki fasilitas yang lengkap untuk para *Businessman.*

2. *Recreational hotel*

   Hotel yang dibuat dengan tujuan untuk orang-orang yang akan santai atau berekreasi.

2.2.4.3 Klasifikasi hotel berdasarkan faktor lokasinya

Klasifikasi hotel berdasarkan lokasinya, dapat dibagi menjadi lima bagian, yaitu :

1. *City hotel*

City hotel adalah hotel yang terletak di dalam kota, dimana sebagian besar tamunya yang menginap melakukan kegiatan bisnis.

2. *Resort hotel*

   Resort hotel adalah hotel yang terletak di kawasan wisata, dimana sebagian besar tamu yang menginap tidak melakukan kegiatan usaha. Macam-macam *resort hotel* berdasarkan lokasi, antara lain :

   a. *Mountain hotel* (hotel yang berada di pegunungan).
   b. *Beach hotel* (hotel yang berada di pinggir pantai).
   c. *Lake hotel* (hotel yang berada di tepi danau).
   d. *Hill hotel* (hotel yang berada di puncak bukit).
   e. *Forest hotel* (hotel yang berada di kawasan hutan lindung).

3. *Suburb hotel*

   Suburb hotel adalah hotel yang lokasinya di pinggiran kota, yang merupakan kota satelit yakni pertemuan antara dua kotamadya.

4. *Urban hotel*

   Urban hotel adalah hotel yang berlokasi di pedesaan dan jauh dari kota besar atau hotel yang terletak di daerah perkotaan yang tadinya hanya berupa desa.

5. *Airport hotel*

Airport hotel adalah hotel yang berada dalam kompleks bangunan atau area pelabuhan udara atau di sekitar Bandar udara.

2.2.4.4 Klasifikasi hotel berdasarkan faktor daya jual dan perencanaan penjualan (*hotel plan usage)*

Hotel plan usage adalah suatu sistem penjualan harga kamar dimana harga kamar yang dijual hanya berupa harga kamar saja atau merupakan sistem harga paket. Beberapa macam *hotel plan usage*, antara lain :

1. *European Plan*

   Biaya yang dikeluarkan untuk menyewa kamar hanya untuk harga kamar saja. Keistimewaan dari *European Plan*, antara lain:

   a. Praktis banyak digunakan di hotel-hotel.
   b. Memudahkan sistem *Billing* (pembayaran pada saat *check-out*).
   c. Semua sistem pemasaran kamar kebanyakan menggunakan sistem ini.

2. *American Plan*

   Sistem perencanaan harga kamar dimana harga yang dibayarkan sudah termasuk harga kamar itu sendiri ditambah dengan harga makan (*meals*).

3. *Continental Plan*

Continental Plan adalah perencanaan harga kamar dimana harga kamar tersebut sudah termasuk dengan *continental breakfast.*

4. *Bermuda Plan*

Bermuda Plan adalah perencanaan harga kamar dimana harga kamar yang dibayar sudah termasuk dengan *American breakfast.*

2.2.4.5 Klasifikasi hotel berdasarkan faktor jumlah kamar dan persyaratan lainnya

Tingkatan hotel didasarkan pada jumlah bintang yang disandang dan jumlah kamar serta persyaratan lainnya dapat diperinci sebagai berikut :

1. Klasifikasi hotel berbintang satu (*)

   Persyaratan :

   a. Jumlah kamar standar, minimum 15 kamar.
   b. Kamar mandi di dalam.
   c. Luas kamar standar, minimum 20 m^2.

2. Klasifikasi hotel berbintang dua (**)

   Persyaratan :

   a. Jumlah kamar standar, minimum 20 kamar.
   b. Kamar suite, minimum 1 kamar.
   c. Kamar mandi di dalam.
   d. Luas kamar standar, minimum 22 m^2.
   e. Luas kamar suite, minimum 44 m^2.

3. Klasifikasi hotel berbintang tiga (***)

   Persyaratan :

   a. Jumlah kamar standar, minimum 30 kamar.

   b. Jumlah kamar suite, minimum 2 kamar.

   c. Kamar mandi di dalam.

   d. Luas kamar standar, minimum 24 m^2.

   e. Luas kamar suite, minimum 48 m^2.

4. Klasifikasi hotel berbintang empat (****)

   Persyaratan :

   a. Jumlah kamar standar, minimum 50 kamar.

   b. Jumlah kamar suite, minimum 3 kamar.

   c. Kamar mandi di dalam.

   d. Luas kamar standar, minimum 24 m^2.

   e. Luas kamar suite, minimum 48 m^2.

5. Klasifikasi hotel berbintang lima (*****)

   Persyaratan :

   a. Jumlah kamar standar, minimum 100 kamar.

   b. Jumlah kamar suite, minimum 4 kamar.

   c. Kamar mandi di dalam.

   d. Luas kamar standar, minimum 26 m^2.

   e. Luas kamar suite, minimum 52 m^2.

2.2.4.6 Klasifikasi hotel berdasarkan ukuran hotel

Klasifikasi hotel berdasarkan ukurannya dapat ditentukan dengan jumlah kamar yang ada. Ukuran hotel diklasifikasikan menjadi 3 bagian, yaitu :

1. *Small hotel*

   Small hotel adalah hotel kecil dengan jumlah kamar di bawah 150 kamar.

2. *Medium hotel*

   Medium hotel adalah hotel dengan ukuran sedang, dimana dalam *medium hotel* ini ada dua kategori, yaitu:

   a. *Average hotel* dengan jumlah kamar antara 150 hingga 299 kamar.

   b. *Above average hotel* dengan jumlah kamar antara 300 hingga 600 kamar.

3. *Large hotel*

   Large hotel adalah hotel dengan klasifikasi sebagai hotel besar degan jumlah kamar minimal 600 kamar.

2.2.4.7 Klasifikasi hotel berdasarkan faktor lamanya tamu menginap

Lamanya tamu menginap di hotel dapat dibagi dalam tiga kategori, yaitu :

1. *Transit hotel*

Tamu yang menginap dalam waktu singkat, rata-rata hanya satu malam.

2. *Semi-residential hotel*

   Tamu yang menginap lebih dari satu malam, tetapi jangka waktu menginap tetap pendek. Kira-kira berkisar antara dua minggu hingga satu bulan.

3. *Residential hotel*

   Tamu yang menginap dalam waktu cukup lama, kira-kira paling sedikit satu bulan.

2.2.4.8 Klasifikasi hotel berdasarkan faktor kegiatan tamu selama menginap

Banyak kegiatan tamu secara spesifik selama menginap di hotel karena dengan maksud-maksud tertentu. Kegiatan-kegiatan tersebut, antara lain :

1. Olahraga
   - *Sport hotel* adalah hotel yang berada pada kompleks kegiatan olahraga.
   - *Ski hotel* adalah hotel dengan menyediakan area sebagai tempat bermain ski. Banyak terdapat di negara yang mempunyai empat musim.
2. Bisnis
   - *Conference hotel* adalah hotel yang menyediakan fasilitas lengkap untuk konferensi.

- *Convention hotel* adalah hotel sebagai bagian dari kompleks kegiatan konvensi.

3. Beribadah
   - *Pilgrim hotel* adalah hotel yang sebagian tempatnya berfungsi sebagai fasilitas beribadah. Seperti hotel-hotel di Arab pada saat musim haji dan Lourdes di Perancis.
4. Berjudi
   - *Casino hotel* adalah hotel yang sebagian tempatnya berfungsi untuk kegiatan berjudi.

2.2.4.9 Klasifikasi hotel berdasarkan pada kriteria jenis tamu

Jenis-jenis tamu yang menginap maksudnya adalah darimana asal-usul mereka menginap dengan latar belakangnya. Dapat diklasifikasikan sebagai berikut :

1. *Family hotel*

   Family hotel adalah tamu yang menginap bersama keluarganya.
2. *Business hotel*

   Business hotel adalah tamu yang menginap para usahawan.
3. *Tourist hotel*

Tourist hotel adalah tamu yang menginap kebanyakan para wisatawan, baik domestic maupun dari luar negeri.

4. *Cure hotel*

   Cure hotel adalah tamu yang menginap dalam proses pengobatan atau penyembuhan dari suatu penyakit.

2.2.5 Penggolongan Kelas Hotel di Indonesia

Penggolongan kelas hotel di Indonesia yang ditetapkan pada tahun 1977 itu ternyata sama dengan klasifikasi hotel yang dipergunakan di negara-negara Perancis, Spanyol, Yordania, Israel, Ceylon, dan Afrika Selatan pada tahun 1972.

Dan sebagai kelanjutan dari Surat Keputusan Menteri Perhubungan Tahun 1977 tersebut, maka pada tahun 1978 Direktur Jendral Pariwisata telah mengeluarkan Surat Keputusan No.Kep.22/U/VI/78 tentang Ketentuan Pelaksanaan Peraturan Usaha dan Klasifikasi Hotel, yang isinya antara lain :

Pasal 13 ayat 1 :

Bahwa setiap sertifikat bagi masing-masing golongan kelas bintang hotel berlaku untuk masa lima tahun dan dapat diperpanjang atau diperbaharui setelah diadakan penilaian kembali oleh Direktur Jendral Pariwisata.

Pasal 14 ayat 1 :

Bahwa sertifikat golongan kelas bintang hotel harus dipasang di Kantor Depan Hotel (*front office*) yang mudah terlihat oleh tamu hotel.

Pasal 14 ayat 2 :

Bahwa hotel-hotel yang telah ditetapkan golongan kelas bintangnya akan dicantumkan dalam "*Indonesia Hotel Directory*", yang akan diterbitkan setiap tahun oleh Direktorat Jendral Pariwisata.

Perlu diketahui pula bahwa golongan kelas (bintang) hotel tersebut masih dapat berubah, naik atau turun, sesuai dengan keadaan dan perkembangan hotel tersebut serta hasil penilaian dari Direktorat Jendral Pariwisata.

Adapun tujuan dari penggolongan kelas (bintang) hotel tersebut adalah :

a. Untuk menjadi pedoman teknis bagi calon investor (penanam modal) di bidang usaha perhotelan.
b. Agar para calon tamu/ penghuni hotel dapat mengetahui fasilitas dan pelayanan yang akan diperoleh di suatu hotel, sesuai dengan golongan kelas/ bintangnya.
c. Agar tercipta persaingan (kompetisi) yang sehat antara sesame pengusaha hotel.
d. Agar tercipta keseimbangan antara permintaan (*demand*) dengan penawaran (*supply*) dalam usaha akomodasi hotel.
e. Dari segi pengelola (*management* dan *staff*), dapat mengangkat nama baik (citra) bila mereka berbisnis atau bekerja pada hotel berbintang.

Jadi, klasifikasi kelas (bintang) hotel yang disebarluaskan itu memang sangat bermanfaat bagi calon investor, para calon tamu, dan

juga bagi para pengelola (*management* dan *staff*) hotel yang bersangkutan.

2.2.6 Reservasi

Reservasi adalah sebuah proses perjanjian berupa pemesanan sebuah produk baik barang maupun jasa dimana pada saat itu telah terdapat kesepahaman antara konsumen dengan produsen mengenai produk tersebut namun belum ditutup oleh sebuah transaksi jual – beli. Pada saat reservasi berlangsung biasanya ditandai dengan adanya proses tukar menukar informasi antara konsumen dan produsen agar kesepahaman mengenai produk dapat terwujud.

Memesan kamar merupakan bagian yang penting ada di *department Front Office* tujuan pemesanan kamar adalah agar tamu memperoleh kepastian akan mendapatkan kamar ketika tiba di hotel. Proses pemesanan kamar dapat dilakukan dengan berbagai cara seperti melalui surat, telepon, telegram, *fax*, ataupun tamu bisa datang langsung ke hotel dan menanyakan langsung kepada resepsionis ada kamar yang kosong atau tidak.

Untuk memudahkan petugas dalam menanggulangi pemesanan kamar maka, dilakukan penggolongan atas pemesanan kamar yaitu:

a. *Confirmed Reservation*

   Pemesanan kamar yang sudah pasti dan telah disetujui pihak hotel yang mana dinyatakan oleh hotel dengan mengirimkan surat berisi jaminan atas penyediaan kamar pada tanggal

tersebut dan tamu akan datang sambil membawa surat jaminan itu.

b. *Guaranteed Reservation*

Pemesanan diperkuat pihak pemesanan dengan jaminan pembayaran jika terjadi *no show reservation* atau pembatalan pesanan pada saat pemesanan. Jaminan tersebut biasanya berupa pembayaran untuk satu malam yang disebut dengan *no show charge.*

c. *Deposit Reservation*

Pemesanan yang disertai pembayaran dimuka untuk satu atau dua hari (malam). Dalam akan menjadi milik hotel jika terjadi *no show reservation.*

d. *Waiting List Reservation*

Pemesanan ini tidak menjamin tamu mendapatkan kamar pada saat yang ditemukan dan pihak hotel akan memberikan kamar jika persediaan kamar masih ada.

e. *Tentative Reservation*

Pemesanan kamar yang tidak pasti dan umumnya dijadikan sebagai catatan saja oleh hotel. Hotel tidak wajib menyediakan kamar untuk pesanan yang bersifat *tentative*.